Deep Learning - Exercise 1

Fall 2024

To do these exercises, you will use Python with 3 and the following packages:

- NumPy. This is highly recommended package to create various data transformation / generate data.
- Pandas. With this package you can import your data into a data frame similar to how it's done in R.
- Matplotlib. This package allows you to graph your data and data transformations.
- PyTorch. This package is a versatile deep learning framework that facilitates building and training of neural networks.
- Seaborn. This package is built on top of Matplotlib, providing a high-level interface for creating informative and aesthetically appealing statistical graphics in Python.

You are not strictly forced to use these packages, but it is highly recommended. Feel free to use other packages you think are necessary.

Exercise 1

Make a simple neural network using PyTorch. The goal of this exercise is to look at the MNIST dataset, and tell what numbers are written. Create a model with 3 fully connected hidden layers, and a 10 unit output layer.

Load the dataset by:

```
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms
import seaborn as sns
import matplotlib.pyplot as plt

# Download the MNIST dataset
transform = transforms.ToTensor()
train_dataset = datasets.MNIST(root='./data', train=True,
download=True, transform=transform)
test_dataset = datasets.MNIST(root='./data', train=False,
download=True, transform=transform)
train_loader = torch.utils.data.DataLoader(dataset=train_dataset,
batch_size=64, shuffle=True)
test_loader = torch.utils.data.DataLoader(dataset=test_dataset,
batch_size=64, shuffle=False)
```

1. Create a neural network:
    a. Initialize 3 layers
    b. Define the **forward** function:
        i. Reshape the data to a fully connected layer. Hint: Use **.view()**.
        ii. Let the input pass through the different layers.
        iii. Consider what activation function you want to use in between the layers, and for the final layer.
    c. Loss function and optimizer:
        i. Consider what loss function and optimizer you want to use.
    d. Create the training loop:
    e. Create the evaluation loop:
    f. Save the model
2. Report your accuracy, is this satisfactory? Why / why not?
3. Plot the loss curves for training as well as test performance.